GPU-Accelerated Population Annealing Algorithm: Frustrated Ising Antiferromagnet on the Stacked Triangular Lattice

Borovsky, M. , Weigel, M. , Barash, L.Y. and Zukovic, M.

EPJ Web of Conferences **108**, 02016 (2016)
DOI: 10.1051/epjconf/201610802016

GPU-Accelerated Population Annealing Algorithm: Frustrated Ising Antiferromagnet on the Stacked Triangular Lattice

Michal Borovský[1,a], Martin Weigel[2,b], Lev Yu. Barash[3,4,c], and Milan Žukovič[1,d]

[1] *Department of Theoretical Physics and Astrophysics, Faculty of Science, P.J. Šafárik University, Park Angelinum 9, 040 01 Košice, Slovak Republic*
[2] *Applied Mathematics Research Centre, Coventry University, Coventry, CV1 5FB, United Kingdom*
[3] *Landau Institute for Theoretical Physics, 142432 Chernogolovka, Russia*
[4] *Science Center in Chernogolovka, 142432 Chernogolovka, Russia*

Abstract. The population annealing algorithm is a novel approach to study systems with rough free-energy landscapes, such as spin glasses. It combines the power of simulated annealing, Boltzmann weighted differential reproduction and sequential Monte Carlo process to bring the population of replicas to the equilibrium even in the low-temperature region. Moreover, it provides a very good estimate of the free energy. The fact that population annealing algorithm is performed over a large number of replicas with many spin updates, makes it a good candidate for massive parallelism. We chose the GPU programming using a CUDA implementation to create a highly optimized simulation.
It has been previously shown for the frustrated Ising antiferromagnet on the stacked triangular lattice with a ferromagnetic interlayer coupling, that standard Markov Chain Monte Carlo simulations fail to equilibrate at low temperatures due to the effect of kinetic freezing of the ferromagnetically ordered chains. We applied the population annealing to study the case with the isotropic intra- and interlayer antiferromagnetic coupling ($J_2/|J_1| = -1$). The reached ground states correspond to non-magnetic degenerate states, where chains are antiferromagnetically ordered, but there is no long-range ordering between them, which is analogical with Wannier phase of the 2D triangular Ising antiferromagnet.

1 Introduction

The use of graphics processing units (GPU) for general purpose computing (GPGPU) is motivated by the fact that the theoretical peak performance of the parallel GPU architecture significantly exceeds the performance of the currently available CPU processors, which can be used to effectively reduce the computational time for a suitable task that can be parallelized. This performance disproportion arises from the fact that the increase in the CPU clock rates has slowed down considerably in the last decade due to the limitations of the used semiconductor technology. Therefore, the focus has been redirected to the multicore solutions. However, even with this approach the performance of the

[a] e-mail: borovsky.michal@gmail.com
[b] e-mail: Martin.Weigel@coventry.ac.uk
[c] e-mail: barash@itp.ac.ru
[d] e-mail: milan.zukovic@upjs.sk

CPUs has been increased by a factor of 16[1]. On the other hand, in the same time frame the single precision performance of the NVIDIA GPUs has grown by an order of two[2]. Of course, the efficiency of completion of a parallel task does not depend only on the sheer performance of the computational unit but also by the way of manipulation with individual types of GPU memories, which differ in size, bandwidth, functionality and locality. The speedup that can be gained as compared to sequential CPU computing, highly depends on our knowledge of the GPU CUDA architecture and the way it executes kernels. It takes a lot of thought and caution to incorporate all of this to create a highly optimized CUDA program.

The use of GPGPU for scientific applications is of interest for instance in the stochastic simulations of spin models [1–4]. Our main goal is to incorporate the GPU-accelerated computing in the population annealing (PA) method proposed by Hukushima and Iba [5]. In section 2, the principles of the PA algorithm are reviewed. The section 3 describes techniques implemented in the creation and optimization of a GPU code of this algorithm. Our second goal is to study the highly frustrated Ising antiferromagnet on the stacked triangular lattice, which suffers from a slow spin dynamics in the low temperature region [6], where standard Markov Chain Monte Carlo (MCMC) simulations fail. This problem will be briefly discussed in the section 4. To deal with this problem we applied PA annealing algorithm on this system and the results will be presented in the section 5.

2 Population annealing algorithm

The PA algorithm was developed to study systems with rough free energy landscapes. The conventional approaches of stochastic statistical physics, such as MCMC algorithm, fail for such systems due to their inability to overcome large energy barriers in the low temperature region. The main strength of the population annealing algorithms lies in the use of a large number R of replicas that undergo the annealing process, starting the replicas at a sufficiently large temperature T (usually at $\beta = (k_B T)^{-1} = 0$), where the energy landscape is quite smooth. The population of replicas helps us to cover the large portion of this surface, so when we reach low temperatures by gradually cooling them, there is a higher probability, that some replicas end up in the global minimum, which corresponds to the searched ground state (GS). However, the effectiveness of this search strongly depends on the ability of these replicas to explore their local neighborhood which brings the entire population to the equilibrium. The population annealing algorithm uses two conceptually different approaches, which are performed at each step $\Delta\beta > 0$ to achieve this. The first of them is a resampling process. Resampling is based on the use of a reweighting technique which moves the population closer to the Gibbs distribution. The chances of a replica to survive and reproduce is proportional to the reweighting factor $e^{-\Delta\beta E_i}$, where E_i is the energy of the i-th replica. Since $\Delta\beta > 0$ the replicas with low energies are more likely to produce copies than high-energy ones, which will probably die out in this process. To keep the population size R more or less unchanged we have to apply a proper normalization factor Q to the reweighting factors, which is in fact the partition function ratio

$$Q = \frac{1}{R} \sum_{i=1}^{R} e^{-\Delta\beta E_i}. \tag{1}$$

Then the expected number of i-th replica's copies is

$$\tau_i = \frac{e^{-\Delta\beta E_i}}{Q}. \tag{2}$$

[1] comparison of Intel Pentium 4 570J (released in 2004) and Intel i7-4790K (2014)
[2] comparison of NVIDIA GeForce 6800 GT (2004) and GTX 980 (2014)

In the ideal case with a statistical ensemble of $R \to \infty$ samples, the resampling process should be enough to obtain the Gibss distribution. However, practically we have only a finite population, so the resampling only redistributes population among the already occupied energy levels, thus leaving the lower energy states poorly sampled. Also making many copies of replicas leaves the population correlated. These two obstacles can be overcome by applying θ MC equilibration sweeps on all replicas. After that we are able to average the observables over replicas, which now should sample more correctly the Gibbs distribution. The free energy estimates can be evaluated from partition function ratios. A detailed discussion of the PA algorithm is given in [7].

3 GPU realization

Considering the fact that parts of the algorithm are performed over large number of replicas ($R \gtrsim 10^4$) it is rather straightforward to assume that the PA algorithm is suitable for massive parallelism. Since the modern GPU architectures contains a few thousand of CUDA cores, we decided to use GPGPU to create an optimized CUDA program for the PA algorithm.

We implemented two levels of parallelism. The first level is done over replicas, where each replica is manipulated by a single thread. Such parallelism is used when we calculate the partition function ratio Q and the normalized weights τ_i. The second and much deeper level is the one that is performed over the spins of each replica, such as internal energy E calculation, magnetization M calculation and parallel checkerboard MC update. In this case one block of threads operates on arrays of spins from one replica, while one block is associated to one replica. To achieve the maximum occupancy of streaming multiprocessors (SMs), we must consider a block size large enough to be partitioned into several warps, but still within the thread-per-block limit, which is specific for each GPU architecture. For instance, we can choose for a 3D system $8 \times 8 \times 8 = 512$ threads per block. However, we have to be very careful not to exceed the register and shared memory limitations for SMs.

The Boltzmann factors in the Metropolis algorithm are tabulated and are implemented on GPU as fetches from a texture. We also used the optimized parallel reduction algorithm presented in [8] for summing Q, M and E. Moreover, we chose the spin arrays to have a block-wise coalescent data pattern to improve the global memory bandwidth.

Another issue of our simulation is to generate parallel long sequences of pseudo-random numbers (PRN) for Metropolis checkerboard spin update and yet its buffer has to fit into the global memory and we also have to think about the performance of a generator. For starters we are using Philox_4x32_10 from the "cuRAND" library, which meets these criteria quite satisfactorily.

4 Stacked triangular Ising antiferromagnet

The stacked triangular Ising antiferromagnet is described by the Hamiltonian

$$H = -J_1 \sum_{\langle i,j \rangle} S_i S_j - J_2 \sum_{\langle i,k \rangle} S_i S_k, \tag{3}$$

where $S_i = \pm 1$ are Ising spin variables, the first term is summed over all intralayer (interchain) couplings with antiferromagnetic interaction $J_1 < 0$ and the second term represents the sum over all interlayer (intrachain) couplings with antiferromagnetic interaction $J_2 < 0$. For simplicity, we will consider the isotropic case with $J_1 = J_2$. Figure 1 describes the topology of this system with a checkerboard system decomposition into six sublattices for the parallel spin update.

The main issue with this system is that when standard MCMC methods are applied the system is unable to reach GS even for a large number of MC sweeps. Such a behavior happens due to the so

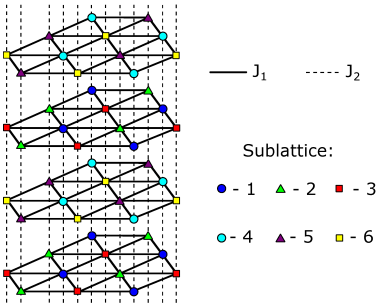

Figure 1. Checkerboard decomposition of the stacked triangular lattice

called kinetic freezing effect [6]. We performed MCMC simulation for a system of $24 \times 24 \times 32$ spins and 10^5 MC sweeps (+20% for equilibration) to demonstrate it. Figure 2(a) depicts the temperature dependence of the internal energy and the heat capacity, which are consistent with the previous results [9]. The primary heat capacity peak shows the transition from the paramagnetic to the partially ordered phase. The round low-temperature peak represents a structural change which is accompanied by the dominance of correlations in the chains, which leads to the gradual antiferromagnetic aligning of the spin chains. However, the inset clearly demonstrates that the system froze in a metastable state, with the energy slightly above the GS $E/N|J_1| = -2$. To observe what happened, we plotted in figure 2(b) the snapshot at $k_BT/|J_1| = 0.01$ of an intrachain staggered magnetization

$$o_z = \sum_{k=1}^{L_z} (-1)^k S_k, \tag{4}$$

where $L_z = 32$ is the number of layers. This parameter reaches saturated values of ± 32 for fully antiferromagnetically ordered chains. As we can see, most of the spin chains are fully ordered with no long-range order between them, which matches a 3D analogue of the Wannier phase [10]. Only one chain highlighted with a circle has the unsaturated value. Its spin configuration is illustrated in the figure 2(c). The energy difference between the GS and this metastable state lies in the presence of the two ferromagnetic couplings, which delimits the chain fragment which has to be flipped entirely in order to get the desired GS. However, the slow spin dynamics at such low temperatures prevents

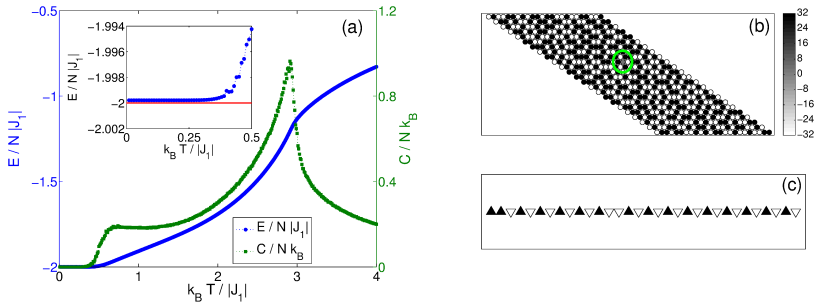

Figure 2. (a) The temperature dependence of the internal energy per spin and the heat capacity per spin. The inset depicts a detail on the internal energy in the low-temperature region. The solid line refers to the GS energy. (b) The snapshot of the intrachain staggered magnetization o_z at $k_BT/|J_1| = 0.01$. The unsaturated chain is marked with a circle. (c) Spin configuration of the selected unsaturated chain.

that to happen. Of course, one can attempt to use multiple-spin-flip updates to resolve this problem [11], in which one of the 2^n states of the n-spin cluster is chosen at each MC trial. In our case, we emphasize the strengths of the PA algorithm to find the GS of our system.

5 Results

We ran three simulations with different sets of parameters on the same lattice size as we used in a MCMC simulation. The first simulation (simulation A) ran on a population of $R = 10^4$ replicas, with $\theta = 100$ MC sweeps and $\Delta\beta = 0.01$ in the range from $\beta = 0$ to 10. In the second one (simulation B) we used twice as finer a step $\Delta\beta = 0.005$ and the third one (simulation C) has ten times the population size compared to simulation B. The obtained results for the internal energy are plotted in figure 3(a). As we can see for all simulations we have successfully converged to the GS configuration, but there is a small difference in the slope of the energy curves. The case with a larger R samples the Gibbs distribution better. Also the smaller temperature step is reducing the bias, because the energy histograms have a larger overlap.

To quantify the equilibration of the population annealing we follow the procedure presented in [12], where Wang et al. calculated the family entropy

$$S_f = -\sum_i \nu_i \ln \nu_i, \tag{5}$$

where the ν_i is the fraction of the population that originates in the i-th replica from the initial population. The e^{S_f} represents the effective number of surviving families. Figure 3(b) shows the family entropy as a function of the temperature for our PA simulations. The first thing that we can observe is that S_f of a population with larger R is larger, which is obvious, because the larger R means reduction of the statistical errors. The figure also shows that S_f substantially drops at the positions of the heat capacity peaks. The effect of the kinetic freezing at the secondary peak drastically diminishes the diversity of the families in the population in contrast to the high-T phase transition. However, this drop is more prominent for the case of a larger population, which does not make much sense, because we were expecting more families to survive in this case. Also the number of different GS observed at the lowest T was 32, 23 and 171 for the simulations A, B and C respectively. The effective number of survived families at the lowest temperature was $e^{S_f} = 1.5857$ for the simulation A, $e^{S_f} = 2.1845$ for

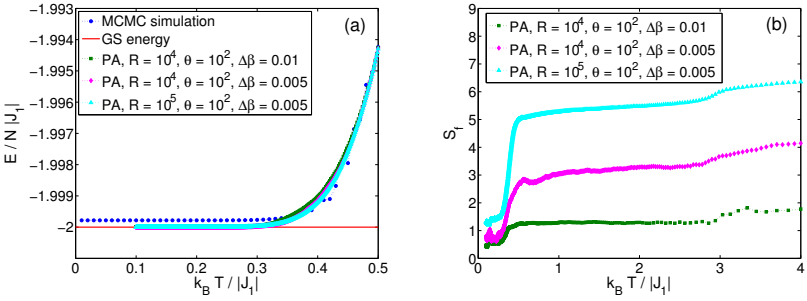

Figure 3. (a) The internal energy comparison of the MCMC (see Fig. 2(a) inset) and PA results in the low-temperature region for system of a size 24x24x32. (b) The family entropy as a function of the temperature for different simulation setups.

the simulation B and $e^{S_f} = 3.7375$ for the simulation C, which is far from the required $e^{S_f} \geq 100$ (or $S_f \gtrsim 4.6$).

The best performance of the PA code what we can get so far was achieved in the simulation C with 0.208 ns per spin-flip on NVIDIA GTX Titan with the speedup up to the 443 times compared to the sequential MCMC code (92.274 ns per spin-flip), which ran on a single core of the Intel i7-4790K processor at 4.4 GHz.

6 Conclusions

We created a GPU-accelerated population annealing algorithm for the stacked triangular Ising antiferromagnet in order to study its ground states. The algorithm converged for all tested sets of simulation parameters. However, the numerical accuracy of averaging was insufficient due to the small number of different states in the population and the rather small family entropy under the secondary heat capacity peak.

There are many possible ways to improve the performance, such as the choice of a more efficient high quality PRN generator, the parallel resampling of replicas in the GPU global memory, the use of asynchronous multispin coding and also by applying an adaptive inverse temperature step based on the fixed overlap of the reweighted energy histograms. We would like to explore these possibilities in the future.

Acknowledgements

M.B. and M.Ž. were supported by the Scientific Grant Agency of Ministry of Education of Slovak Republic (Grant No. 1/0331/15) and M.B. also by the Faculty of Science UPJŠ (Grant ID. VVGS-PF-2015-490). M.W. and L.Yu.B. acknowledge support from the European Commission through the IRSES network DIONICOS under Contract No. PIRSES-GA-2013-612707. L.Yu.B. also acknowledges support by the Russian Science Foundation grant No. 14-21-00158.

References

[1] T. Preis, P. Virnau, W. Paul, and J. J. Schneider, J. Comput. Phys. **228**, 4468–4477 (2009)
[2] M. Weigel, J. Comput. Phys. **231**, 3064–3082 (2012)
[3] Y. Komura, and Y. Okabe, Comput. Phys. Commun. **183**, 1155–1161 (2012)
[4] Y. Fang, S. Feng, K.-M. Tam, Z. Yun, J. Moreno, J. Ramanujam, and M. Jarrell, Comput. Phys. Commun. **185**, 2467–2478 (2014)
[5] K. Hukushima, and Y. Iba, AIP Conf. Proc. **690**, 200–206 (2003)
[6] R. R. Netz, and A. N. Berker, Phys. Rev. Lett. **66**, 377–380 (1991)
[7] J. Machta, Phys. Rev. E **82**, 026704 (2010)
[8] D. B. Kirk, and W.-M. W. Hwu, *Programming Massively Parallel Processors: A Hands-on Approach* (Morgan Kaufmann, Burlington, USA, 2010) 102–103
[9] D. Blankschtein, M. Ma, A. N. Berker, G. S. Grest, and C. M. Soukoulis, Phys. Rev. B **29**, 5250–5252 (1984)
[10] M. Žukovič, L. Mižišin, and A. Bobák, Acta Physica Polonica A **126**, 40–41 (2014)
[11] J.-J. Kim, Y. Yamada, and O. Nagai, Phys. Rev. B **41**, 4760–4763 (1990)
[12] W. Wang, J. Machta, and H. G. Katzgraber, Phys. Rev. E **92**, 013303 (2015)